REMARKS

Claims 1-3, 5-7, 9, 11-14, and 16-19 are pending.

Claims 4, 8, 10, 15, and 20-26 have been cancelled.

Claims 27-31 have been added.

In the Office Action mailed September 28, 2009, claims 1-26 were rejected under 35 U.S.C. § 102(a) as anticipated by Kuno (Conversions + Interferences = Business Logic).

In response to the Decision on Appeal dated April 28, 2009, in which the Final Rejection dated December 7, 2006, was reversed, a new § 102 rejection over Kuno has been raised in the present Office Action.

It is respectfully submitted that the teachings of Kuno are different from the claimed subject matter. To clarify the distinctions, claim 1 has been amended to recite that the conversation logic repository includes plural conversation logic. Claim 1 further recites dynamically discovering a service associated with a node (of a workload definition) with no hard-coded conversation logic, where the discovered service is selected from among plural services. Claim 1 further recites selecting one of the plural conversation logic in the conversation logic repository based on the discovered service. Support for the amendments of claim 1 can be found at least in the following passages of the present application: page 9, lines 1-17; page 10, lines 1-9, 19-26; page 17, line 23 – page 20, line 2.

Note that claim 1 specifically recites dynamically discovering a service (selected from among plural services) associated with the node of a workload definition with no hard-coded conversation logic, in combination with selecting one of plural conversation logic based on a discovered service. In the rejection of claim 1, the Office Action equated a "conversation definition" as disclosed in Kuno with the "conversation logic" recited in claim 1. The "node" of the workload definition with no hard-coded conversation logic of claim 1 was equated by the Office Action with the "client" of Kuno. Moreover, the Office Action cited Section 4 of Kuno as disclosing mapping a conversation to an "appropriate service." 09/28/2009 Office Action at 3.

It is respectfully submitted that Kuno does not disclose **first** discovering a service from among plural services that is associated with the node with no hard-coded conversation logic that is executed, **followed** by selecting one of plural conversation logic **based on the discovered service**. Section 4 of Kuno refers to a dynamic conversation controller for e-services. The

conversation controller of Kuno acts as a proxy to a service. Kuno, Section 4, ¶ 2. Once the conversation controller receives a message on behalf of an e-service, the conversation controller dispatches the message to the appropriate entry point of the service, based on the state of the conversation and the type of document. *Id.* When forwarding the response from an e-service to a client, the conversation controller includes a prompt indicating valid document types that are accepted by the next stage of the conversation. *Id.*, ¶ 3. The conversation controller can also direct the client's side of the conversation. *Id.*

In fact, and as explained by Kuno, a service is considered to support a conversation definition. *Id.*, Section 3.2, ¶ 3 ("A service that supports this conversation definition expects a conversation to begin with the receipt of a LoginRQ or a RegistrationRQ document."). As further explained by Kuno, "a client could use the same conversation specification to talk to two different book-selling services . . .." *Id.*, Section 2, ¶ 6. Details regarding tasks performed by the conversation controller in cooperation with a client in an e-service are provided in Fig. 5 on page 11 of Kuno. As shown in Fig. 5 of Kuno, the conversation controller looks at a message header and determines the current state of the conversation (which assumes that the conversation definition is already currently being used by the conversation controller). From **the** conversation specification (or conversation definition), valid input document types for a current state are obtained, and it is determined whether the current message is of a valid input document type for the current state. *Id.*, Fig. 5, steps 2-3. If the received message is of a valid type, then the inbound document is looked up in the dispatch specification and the message is dispatched to the appropriate service entry point; otherwise, the client is informed that the message is not a valid type. *Id.*, step 4. Nowhere in Fig. 5 of Kuno, which describes the tasks performed by the conversation controller, is there any hint of first dynamically discovering a service (selected from among plural services) associated with the node with no hard-coded conversation logic (of a workload definition), followed by selecting one of plural conversation logic in the conversation logic repository **based on the discovered service**.

Moreover, note that claim 1 further recites dynamically plugging in the selected conversation logic into the node of the workload definition at runtime. The Office Action cited Section 4.1 of Kuno as purportedly disclosing this feature of claim 1. 09/28/2009 Office Action at 3. Section 4.1 of Kuno refers to client automation, which refers to decoupling conversation

logic from business logic on the client side to increase flexibility of the client by allowing the client to interact dynamically with services even if the conversation policies do not match exactly. Section 4.1 of Kuno also states that the conversation controller is able to dispatch messages the client receives from a server. The ability of a client to communicate with a conversation controller does not provide any hint of dynamically **plugging in** the selected conversation logic into the node of the workload definition at runtime.

In view of the foregoing, it is respectfully submitted that claim 1 is not anticipated by Kuno.

Independent claim 3 is also not anticipated by Kuno. As purportedly disclosing the "selecting" element of claim 3, the Office Action cited Section 4 of Kuno, ¶¶ 1-3. This passage of Kuno refers to a dynamic conversation controller for e-services discussed above in connection with claim 1. The conversation controller acts as a proxy to an e-service, and tracks the state of an ongoing conversation based on the types of messages exchanged. Kuno, Section 4, ¶ 1. Once the conversation controller receives a message on behalf of an e-service, the conversation controller dispatches the message to the appropriate service entry point based on the state of the conversation and the document's type. There is absolutely no teaching in this passage of Kuno regarding selecting a conversation logic from among plural conversation logic in the conversation logic repository based on the returned service identifier that corresponds to a selected service, where the selected service is selected from among plural services in response to sending of a service selection query to an electronic services platform or other service broker.

Claim 3 is therefore also not anticipated by Kuno.

Independent claim 11 is allowable for at least some of the reasons stated above with respect to claim 1.

Dependent claims, including newly added dependent claims 27-31, are allowable for at least the same reasons as corresponding independent claims.

The Commissioner is authorized to charge any additional fees and/or credit any overpayment to Deposit Account No. 08-2025 (10010118-1).

Respectfully submitted,

Date: December 22, 2009

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883